Exp No. 1 (ALU 8 Bit)

VHDL CODE:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
useieee.NUMERIC_STD.all;
----------------------------------------------
---------- ALU 8-bit VHDL ---------------------
----------------------------------------------
entity ALU is
generic (
constant N: natural := 1  -- number of shited or rotated bits
   );

   Port (
   A, B     : in  STD_LOGIC_VECTOR(7 downto 0);  -- 2 inputs 8-bit
ALU_Sel  : in  STD_LOGIC_VECTOR(3 downto 0);  -- 1 input 4-bit for selecting function
ALU_Out   : out  STD_LOGIC_VECTOR(7 downto 0); -- 1 output 8-bit
Carryout : out std_logic        -- Carryout flag
   );
end ALU;
architectureBehavioral of ALU is

signalALU_Result : std_logic_vector (7 downto 0);
signaltmp: std_logic_vector (8 downto 0);

begin
process(A,B,ALU_Sel)
begin
case(ALU_Sel) is
```

```
when "0000" => -- Addition
ALU_Result<= A + B ;
when "0001" => -- Subtraction
ALU_Result<= A - B ;
when "0010" => -- Multiplication
ALU_Result<= std_logic_vector(to_unsigned((to_integer(unsigned(A)) *
to_integer(unsigned(B))),8)) ;
when "0011" => -- Division
ALU_Result<= std_logic_vector(to_unsigned(to_integer(unsigned(A)) /
to_integer(unsigned(B)),8)) ;
when "0100" => -- Logical shift left
ALU_Result<= std_logic_vector(unsigned(A) sll N);
when "0101" => -- Logical shift right
ALU_Result<= std_logic_vector(unsigned(A) srl N);
when "0110" => --  Rotate left
ALU_Result<= std_logic_vector(unsigned(A) rol N);
when "0111" => -- Rotate right
ALU_Result<= std_logic_vector(unsigned(A) ror N);
when "1000" => -- Logical and
ALU_Result<= A and B;
when "1001" => -- Logical or
ALU_Result<= A or B;
when "1010" => -- Logical xor
ALU_Result<= A xor B;
when "1011" => -- Logical nor
ALU_Result<= A nor B;
when "1100" => -- Logical nand
ALU_Result<= A nand B;
when "1101" => -- Logical xnor
ALU_Result<= A xnor B;
when "1110" => -- Greater comparison
```

```
if(A>B) then
ALU_Result<= x"01" ;
else
ALU_Result<= x"00" ;
end if;
when "1111" => -- Equal comparison
if(A=B) then
ALU_Result<= x"01" ;
else
ALU_Result<= x"00" ;
end if;
when others =>ALU_Result<= A + B ;
end case;
end process;
ALU_Out<= ALU_Result; -- ALU out
tmp<= ('0' & A) + ('0' & B);
 Carryout <= tmp(8); -- Carryout flag
endBehavioral;
```

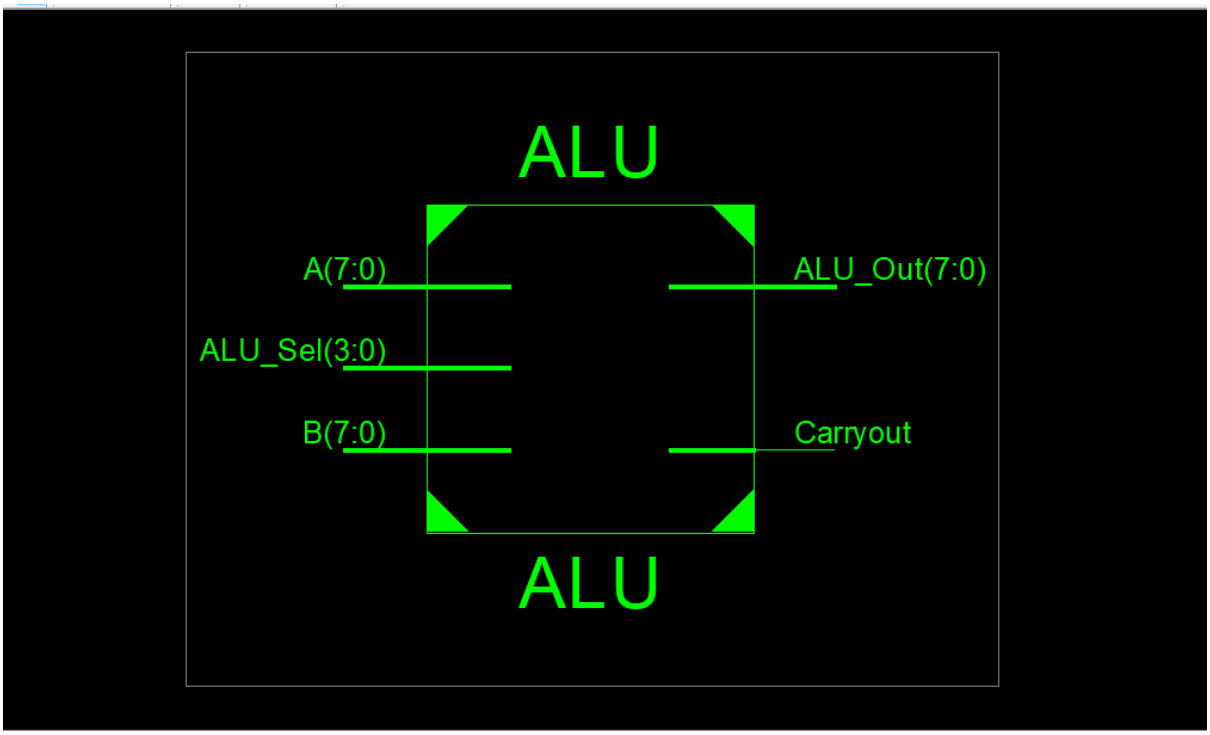
Test Bench Code:

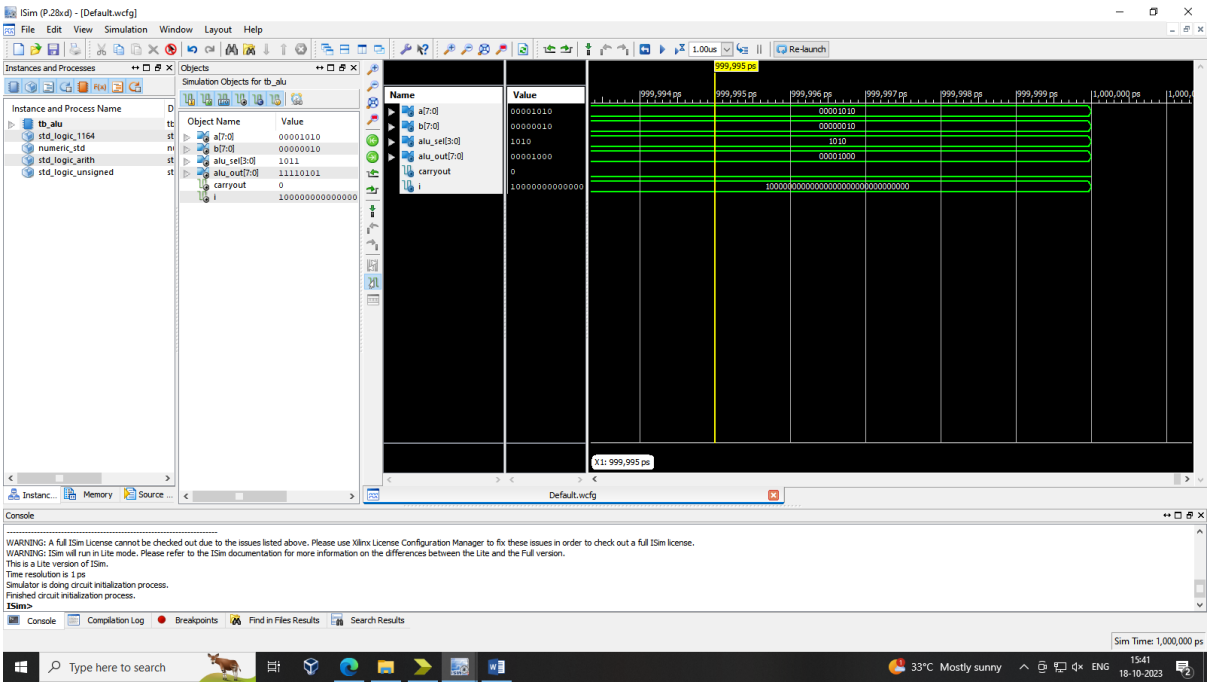
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
useIEEE.std_logic_unsigned.all;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY tb_ALU IS
```

```
END tb_ALU;

ARCHITECTURE behavior OF tb_ALU IS

   -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT ALU
PORT(
A : IN  std_logic_vector(7 downto 0);
B : IN  std_logic_vector(7 downto 0);
ALU_Sel : IN  std_logic_vector(3 downto 0);
ALU_Out : OUT  std_logic_vector(7 downto 0);
Carryout : OUT  std_logic
       );
   END COMPONENT;
      --Inputs
signal A : std_logic_vector(7 downto 0) := (others => '0');
signal B : std_logic_vector(7 downto 0) := (others => '0');
signalALU_Sel : std_logic_vector(3 downto 0) := (others => '0');
--Outputs
signalALU_Out : std_logic_vector(7 downto 0);
signal Carryout : std_logic;

signal i:integer;
BEGIN

 -- Instantiate the Unit Under Test (UUT)
uut: ALU PORT MAP (
        A => A,
        B => B,
ALU_Sel =>ALU_Sel,
```

```
ALU_Out =>ALU_Out,
          Carryout => Carryout
        );
   -- Stimulus process
stim_proc: process
begin
      -- hold reset state for 100 ns.
      A <= x"0A";
  B <= x"02";
ALU_Sel<= x"0";

fori in 0 to 15 loop
ALU_Sel<= ALU_Sel + x"1";
wait for 100 ns;
end loop;
      A <= x"F6";
  B <= x"0A";
wait;
end process;

END;
```

Output:


ALU
A(7:0)
ALU_Out(7:0)
ALU_Sel(3:0)
B(7:0)
Carryout
ALU


ISim (P.28xd) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Objects
Simulation Objects for tb_alu
tb_alu
std_logic_1164
numeric_std
std_logic_arith
std_logic_unsigned
Object Name
Value
a[7:0]
b[7:0]
alu_sel[3:0]
alu_out[7:0]
carryout
i
00001010
00000010
1010
00001000
0
100000000000000
999,995 ps
X1: 999,995 ps
Default.wcfg
Console
WARNING: A full ISim License cannot be checked out due to the issues listed above. Please use Xilinx License Configuration Manager to fix these issues in order to check out a full ISim license.
WARNING: ISim will run in Lite mode. Please refer to the ISim documentation for more information on the differences between the Lite and the Full version.
This is a Lite version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
33°C Mostly sunny
ENG
15:41
18-10-2023